COLUMN シリコンバレーに見る 技術者の職種とスキル

第2回

EDA/CAD エンジニア

東原朋成

LSI設計に携わると，遅かれ早かれ上流工程（あるいはフロントエンド）や下流工程（あるいはバックエンド）ということばに接するようになります．おおまかに分けると，論理合成以降のプロセスを下流工程（もしくはバックエンド工程）と言うようです（論理合成までを上流工程とすることもある）．

● 下流工程はアウトソース

近年，下流工程のアウトソーシングが盛んです．インドには，下流工程を専門にする会社がたくさんあると聞きます．シリコンバレーでは，スタートアップ企業のみならず大手企業もアウトソーシングを多用しています．筆者の知る企業の中には，下流工程のすべて（検証からマスク作成にいたるまで）をアウトソーシングするような例もあります．この企業は，スタートアップ企業（一般にキャッシュ・フローがタイトで投資にすこぶる敏感）というわけではありません．長年関係のあるアウトソーシングの会社に，徐々に設計工程を移していった結果だと言います．マスク作成の際にレビューを実施するほかは，ほとんどすべてをアウトソーシング先にゆだねているそうです．筆者のように古風なエンジニアにはちょっと怖くてできないことです．

このような背景によるものか，ひところ不足だと言われた下流工程を専門とするエンジニアの需要が少なくなっていると聞くことがあります．そのうわさに根拠があるのか確かめたことはありませんが，会社やプロジェクト・マネージャの要求するEDA/CADエンジニアの質が変わってきたのは事実なようです．

● ツールの使い手のしごとはなくなる

以前はEDAエンジニアというと，EDAツールを熟知して，スクリプトを作成し，設計ゴール（おもに速度と規模）を満たすことを専門としている人のことでした．EDAベンダからすると，いちばん高いツールを使ってくれるエンジニアです．セミナや講習会を盛んに開いて，EDAエンジニアやマネージャたちの囲い込みをしたものです．

ところが最近は，ツールの使い手だけのEDA/CADエンジニアのしごとは，アウトソーシング企業に奪われつつあります．筆者の知るEDA/CADエンジニアの中には，Verilog HDLのコードすら満足に理解できないCADツールの使い手もいました．あるEDAベンダのツールには非常に熟知していたのですが，ダウンサイジング（会社規模の縮小や人員削減）の際にしごとを失ってしまいました．今はアウトソーシング企業でコンサルタント（米国の顧客とインドの設計センタをつなぐ窓口）をしています．

● 設計エンジニアとしてのEDA/CADエンジニアが求められる

最近は上流工程と下流工程を有機的に結びつける，アウトソーシング先とも緊密に連係プレーができるEDA/CADエンジニアが求められています．アウトソーシングが盛んといえども，自前のツールで設計をする会社やプロジェクトも健在です．そういった会社やグループは，非常に野心的な仕様やスケジュールで開発を行っています．とはいえ，野心的なプロジェクトに割り当てるエンジニアの数は激減しています．ひところ15人程度の体制で臨んだプロジェクトでも，今では5人程度というのが普通です．オーバヘッドの比較的多い大手企業でも10人以下というのが現状です．

上流工程からの設計を下請けのようにこなしていた下流工程のエンジニア（こういったエンジニアを時としてテクニシャンと呼ぶ）のポジションはなくなりつつあります．しかし，逆に設計エンジニアとしてのEDA/CADエンジニアのポジションができているといってよいのかもしれません．プロジェクトの初期の段階から関与し，場合によってはアーキテクチャ設計にまでかかわってきます．RTL（resistor transfer level）で設計したり，設計の機能検証に直接かかわるわけではありませんが，使用するプロセス，ライブラリ，ツールなどからくる設計への直接的なアドバイスを与えるといったコンサルタントを行います．従来の下流工程を引き受ける社内部門といった位置づけではなく，プロジェクト・メンバの一人といった位置づけです（2～3のプロジェクトを掛け持ちするようなこともある）．小さい会社では，一人で下流工程のすべてを担当することも求められます．特定のツールに詳しいということではなく，すべての工程について詳しくあることも求められます．

さて，このようなEDA/CADエンジニアがいるのかということになりますが，もちろん探し出すことはかなり難しいのが現状です．こういったエンジニアを抱えない（探せない）会社，マネージャは従来の下流工程をどんどんアウトソーシングしていくというのが現状となっています．

前回の回答

（2006年10月号，pp.148-149）

前回の問題は，いずれもプログラミングの実質的な初歩事項にあたります．

プログラミング言語の構文を学ぶことが初歩以前だとすると，データ・タイプを学ぶのが初歩事項にあたります．

前回の問題はともにコンパイラがスタックやレジスタなどをどうアロケートするのかということに主眼を置いています．

こういった点は，教科書ではあまり説明されていません．C/C＋＋のプログラミング・ヒントブックには数々の事例が載っているものです．

問題1

次のプログラムを，組み込みソフトウェア開発エンジニアの観点から批評してください．

```
sub  (int arg)
  {
  char   big_arrary[1024];
  int    i;
  ...
  if  (arg>100)
      {
      for  (i=0; i<1024; i++)
           big_array[i] = 0;
      ...
      }
   }
```

問題1では，配列の宣言の場所を検討します．

問題で示したコードのように配列を宣言した場合，配列はかならず確保されますが，if文の条件が満たされないときには宣言した配列が使用されません．このような場合は，配列の宣言自体を条件が満たされた場合だけにしたほうがいいのかどうか検討する必要があります．

また，配列の宣言がされた時点でスタックに影響がないでしょうか．組み込みシステムのように，メモリ容量に制約がある場合には，スタックに注意を払うことも必要になります．

問題2

次のプログラムを，組み込みソフトウェア開発エンジニアの観点から批評してください．

```
sub  (int* value)
{

  int flag;

   ...

   flag = *value;
   ...

   ...

   i = flag;  // flag はここで初めて使用された

}
```

問題2では，ローカル変数への代入のタイミング（ポジション）を検討してみましょう．

コンパイラの最適化アルゴリズムは非常に進歩しています．しかしながら，ポインタの取り扱いなどはいつも保守的です．ポインタ指定の代入の場合，そのポインタで指定されたアドレスが後々まで変化がないとエンジニアは知っていても，コンパイラには知る余地がありません．このため，とりあえずプログラムの指定どおりローカル変数に保存してしまいますが，このために不必要にスタック・アクセスが発生する可能性が出てしまいました．

今月の問題

問題

EDAツールを使って論理合成することを考えます．EDAツールを最適に使用するためには，ツール（EDAベンダ）ごとにスクリプトを用意する必要があります．設計エンジニアの立場で，どのようなデータ（スクリプトではない）を用意するかについて，いくつかの事例を挙げて説明してください．

とうはら・ともなり